

RRRRRRRRRRRR		MMM		MMM	SSSSSSSSSSSS
RRRRRRRRRRRR		MMM		MMM	SSSSSSSSSSSS
RRRRRRRRRRRR		MMM		MMM	SSSSSSSSSSSS
RRR	RRR	MMMMMM	MMMMMM	SSS	
RRR	RRR	MMMMMM	MMMMMM	SSS	
RRR	RRR	MMMMMM	MMMMMM	SSS	
RRR	RRR	MMM	MMM	SSS	
RRR	RRR	MMM	MMM	SSS	
RRR	RRR	MMM	MMM	SSS	
RRRRRRRRRRRR		MMM		SSSSSSSSSS	
RRRRRRRRRRRR		MMM		SSSSSSSSSS	
RRRRRRRRRRRR		MMM		SSSSSSSSSS	
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM		SSSSSSSSSSSS	
RRR	RRR	MMM		SSSSSSSSSSSS	
RRR	RRR	MMM		SSSSSSSSSSSS	

SY

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

PI

```

LL          IIIII
LL          IIIII
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LLLLLLLLLL IIIII
LLLLLLLLLL IIIII
SSSSSSSS
SSSSSSSS
SS
SS
SS
SS
SSSSSS
SSSSSS
SS
SS
SS
SS
SSSSSSSS
SSSSSSSS

```


(1) 72
(2) 97
(4) 244
(6) 370
(8) 676
(9) 724

DECLARATIONS
RMSSEQJNL - Sequential journaling setup
MAKE_AI_JNL - Put operation specific info in AI jnl
MAKE_BI_JNL - Put operation specific info in BI jnl
CHANGE_BUFF - get next buffer
WRTBIJNL - writes BI/RU journal entry


```
0000 1
0000 2 $BEGIN RM1JOURNL,000,RMSRMS_JOURNAL,<Sequential specific journaling>
0000 3
0000 4 *****
0000 5
0000 6 *
0000 7 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9 * ALL RIGHTS RESERVED.
0000 10 *
0000 11 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16 * TRANSFERRED.
0000 17 *
0000 18 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20 * CORPORATION.
0000 21 *
0000 22 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24 *
0000 25 *****
0000 26
0000 27
0000 28 ++
0000 29
0000 30 FACILITY: RMS-32
0000 31
0000 32 ABSTRACT: This module contains the routines which journal record
0000 33 operations performed on sequential files.
0000 34
0000 35
0000 36 ENVIRONMENT: VAX/VMS Operating System
0000 37
0000 38
0000 39 --
0000 40
0000 41 AUTHOR: Tamar Krichevsky, CREATION DATE: 28-May-1983
0000 42
0000 43 MODIFIED BY:
0000 44
0000 45 V03-005 TSK0004 Tamar Krichevsky 9-Dec-1983
0000 46 Add support for BI journaling.
0000 47
0000 48 *****
0000 49 *
0000 50 * THE CODE FOR BI JOURNALING OF TRUNCATE OPERATIONS HAS NOT
0000 51 * BEEN TESTED.
0000 52 *
0000 53 *****
0000 54
0000 55 V03-004 JWT0141 Jim Teague 11-Nov-1983
0000 56 Change IFBSV_RUM to IFBSV_ONLY_RU
0000 57
```



```

0000 58 : V03-003 TSK0003 Tamar Krichevsky 5-Oct-1983
0000 59 : Use RM$RETJNLBDB and RM$ALJNLBDB instead of RM$RETBDB and
0000 60 : RM$ALBDB when allocating a large journal buffer.
0000 61 :
0000 62 : V03-002 TSK0002 Tamar Krichevsky 27-Jun-1983
0000 63 : Pass journal BDB to RM$WRTJNL instead of related BDB for
0000 64 : AI operations.
0000 65 :
0000 66 : V03-001 TSK0001 Tamar Krichevsky 22-Jun-1983
0000 67 : Clean up comments for MAKE_AI_JNL.
0000 68 :
0000 69 : **
0000 70 :
0000 71 :
0000 72 : .SBTTL DECLARATIONS
0000 73 :
0000 74 : INCLUDE FILES:
0000 75 :
0000 76 :
0000 77 :
0000 78 : MACROS:
0000 79 :
0000 80 : $IFBDEF
0000 81 : $BDBDEF
0000 82 : $IRBDEF
0000 83 : $FABDEF
0000 84 : $RABDEF
0000 85 : $RMSDEF
0000 86 : $RJRDEF
0000 87 : $CJFDEF
0000 88 :
0000 89 : EQUATED SYMBOLS:
0000 90 :
0000 91 :
0000 92 :
0000 93 : OWN STORAGE:
0000 94 :
0000 95 :

```

```
0000 97 .SBTTL RM$SEQJNL - Sequential journaling setup
0000 98
0000 99 :++
0000 100 :
0000 101 : FUNCTIONAL DESCRIPTION:
0000 102 :
0000 103 : RM$SEQJNL is called when a sequential file record operation needs
0000 104 : to be journaled. It fills in the recover journal record (RJR) with
0000 105 : the appropriate information and the returns to the caller.
0000 106 :
0000 107 : CALLING SEQUENCE:
0000 108 :
0000 109 : BSBW RM$SEQJNL
0000 110 :
0000 111 : INPUT PARAMETERS:
0000 112 :
0000 113 : 4(SP) type of record operation to be performed
0000 114 : R4 BDB address
0000 115 : R5 Data record address
0000 116 : R6 Data record size
0000 117 : R8 RAB
0000 118 : R9 IRAB
0000 119 : R10 IFAB
0000 120 : R11 Impure
0000 121 :
0000 122 : IMPLICIT INPUTS:
0000 123 :
0000 124 : IRB$L_JNLBDB Address of journal BDB
0000 125 :
0000 126 : OUTPUT PARAMETERS:
0000 127 :
0000 128 : R0 Status
0000 129 : R1 - R3 Destroyed
0000 130 :
0000 131 : IMPLICIT OUTPUTS:
0000 132 :
0000 133 : None
0000 134 :
0000 135 : COMPLETION CODES:
0000 136 :
0000 137 : None
0000 138 :
0000 139 : SIDE EFFECTS:
0000 140 :
0000 141 : The journal buffer and BDB may be released and new ones allocated,
0000 142 : if the the existing buffer is not large enough to hold the current
0000 143 : record.
0000 144 :
0000 145 :--
0000 146 :
0000 147 :
0000 148 RM$SEQJNL::
54 30 30 BB 0000 149 PUSHR #^M<R4, R5> ; Save BDB and record addresses
30 A9 D0 0002 150 MOVL IRB$L_JNLBDB(R9), R4 ; Get the journal BDB
0006 151
0006 152 :+
0006 153 :
```



```
0006 154 ; The buffer for the journal entry must be large enough to hold the current
0006 155 ; record and any overhead necessary to describe the record. If the buffer is
0006 156 ; not large enough, then deallocate it and its BDB. Then, allocate a new buffer
0006 157 ; (and buffer descriptor block) which will be large enough.
0006 158 ;
0006 159 ; If the operation being journaled is a BI $TRUNCATE or BI $PUT, with the TPT
0006 160 ; option, then skip this check. Journaling for these operations is done block
0006 161 ; mode. Therefore, a different set of criteria is used to check the size of
0006 162 ; the journal buffer.
0006 163 ;
0006 164 ; -
0006 165 ;
0006 166 BBC #IFB$V_BI, IFB$B_JNLFLG(R10), 10$ ; If not BI, check jnl buff size
0006 167 BITB #<RJR$TPT!RJR$_TRUNCATE>, 4(SP) ; BI TPT or TRUNCATE?
0006 168 BNEQ 30$ ; Yes, skip size check
0006 169
0006 170 10$: SUBW3 #RJR$C_RECLEN,BDB$W_ALLOC_SIZE(R4),R1 ; Ignore jnl entry overhead
0006 171 SUBW3 IRB$W_ROVHDSZ(R9), = ; Ignore the overhead for the
0006 172 IRB$W_RTOTLSZ(R9), R2 ; current record, also
0006 173 ADDW2 IFB$B_FSZ(R10), R2 ; Do count fixed header part
0006 174 CMPW R2, RT ; Will record fit in buffer?
0006 175 BLEQU 30$ ; Yes, Make and write jnl entry
0006 176
0006 177 PUSHR #*M<R2> ; Save the record size
0006 178 JSB RM$RETJNLBDB ; Release this buffer and BDB
0006 179 POPR #*M<R5> ; Get record size - It is put in
0006 180 ; R5 because of RM$ALDBUF
0006 181 ADDL2 #RJR$C_RECLEN, R5 ; Add in journal entry overhead
0006 182 ADDL2 #511, R5 ; Round up to a
0006 183 BICL2 #511, R5 ; page boundary
0006 184 JSB RM$ALDJNLBUF ; Get new buffer and BDB
0006 185 BLBS R0, 20$ ; Keep going if everything is ok
0006 186 BRW EXIT ; Get out on error
0006 187 20$: SSB #BDB$V_PRM, BDB$B_FLGS(R4) ; Mark BDB as permanent
0006 188 CLRQ @BDB$L_ADDR(R4) ; Clear the top of the buffer
0006 189 MOVL R4, IRB$L_JNLBDB(R9) ; Save the BDB's address
0006 190 MOVL 4(SP), R5 ; Restore the record address
0006 191 BRB 40$
0006 192
0006 193 ; +
0006 194 ;
0006 195 ; The journal buffer is all set. Now, the journal entry is filled in and
0006 196 ; written to the journal. First, fill in the overhead which describes the
0006 197 ; record.
0006 198 ;
0006 199 ; -
0006 200 ;
0006 201 30$: MOVL BDB$L_ADDR(R4), R3 ; Get the jnl buffer address
0006 202 40$: MOVB #RJR$C_RECORD, RJR$B_ENTRY_TYPE(R3) ; A record is being jnl'ed
0006 203 MOVB #RJR$C_SEQ, RJR$B_ORG(R3) ; File org. is sequential
0006 204 MOVB 12(SP), RJR$B_OPER(R3) ; Type of record operation
0006 205 MOVL IRB$L_RP_VBN(R9), RJR$L_RFA0(R3) ; RFA is needed, get VBN part
0006 206 MOVW IRB$L_RP_OFF(R9), RJR$W_RFA4(R3) ; and offset into block
0006 207 MOVAB RJR$T_RIMAGE(R3), R1 ; Point to start of record image
0006 208
0006 209 ; +
0006 210 ;
```

06 00A0 CA 02 E1 0006 166
04 AE 1F 93 000C 167
53 12 0010 168
51 2C A4 0048 8F A3 0012 169
52 66 A9 64 A9 A3 0019 171
52 5F AA A0 001F 172
51 52 B1 0023 173
3D 1B 0026 174
04 BB 0028 175
00000000 EF 16 002A 176
20 BA 0030 177
55 00000048 8F C0 0032 178
55 000001FF 8F C0 0039 179
55 000001FF 8F CA 0040 180
00000000 EF 16 0047 181
03 50 E8 004D 182
005A 31 0050 183
18 B4 7C 0058 184
30 A9 54 D0 005B 185
55 04 AE D0 005F 186
04 11 0063 187
0065 188
0065 189
0065 190
0065 191
0065 192
0065 193
0065 194
0065 195
0065 196
0065 197
0065 198
0065 199
0065 200
53 18 A4 D0 0065 201
03 A3 02 90 0069 202
04 A3 00 90 006D 203
05 A3 0C AE 90 0071 204
40 A3 48 A9 D0 0076 205
44 A3 4C A9 B0 007B 206
51 48 A3 9E 0080 207
0084 208
0084 209
0084 210


```
0084 211 ; Do the journal-type specific stuff.
0084 212 ;
0084 213 ; AI journaling
0084 214 ;
0084 215 ; -
0084 216
13 00A0 CA 03 E1 0084 217 BBC #IFBSV_AI, IFBSB_JNLFLG(R10), 50$; If not AI jnl'ing, keep going
      24 10 008A 218 BSBB MAKE_AI_JNL ; Make the AI record image.
      1E 50 E9 008C 219 BLBC R0, EXIT ; Get out on error
      54 DD 008F 220 PUSHL R4 ; Use jnl BDB as related BDB
      7E 03 9A 0091 221 MOVZBL #CJFS_AI, -(SP) ; Pass jnl type to RMSWRTJNL
      FF 69 30 0094 222 BSBW RMSWRTJNL ; Write journal entry
      5E 08 C0 0097 223 ADDL2 #8, SP ; Remove arguments from stack
      10 50 E9 009A 224 BLBC R0, EXIT ; Get out on error
      009D 225
      009D 226 ; +
      009D 227 ;
      009D 228 ; BI journaling
      009D 229 ;
      009D 230 ; -
      009D 231
00A0 CA 03 93 009D 232 50$: BITB #<IFBSV_BI!IFBSV_RU>, IFBSB_JNLFLG(R10) ; BI or RU jnl'ing?
      09 13 00A2 233 BEQL EXIT ; No, then continue
      00A4 234
05 A3 1F 93 00A4 235 BITB #<RJR$_TPT!RJR$_TRUNCATE>, RJR$_OPER(R3) ; BI TPT or TRUNCATE?
      03 12 00A8 236 BNEQ EXIT ; Yes, jnl entry was already written
      020D 30 00AA 237 BSBW WRTBIJNL ; Write jnl entry
      00AD 238
      00AD 239 EXIT:
      30 BA 00AD 240 POPR #^M<R4, R5>
      05 00AF 241 RSB
```



```
00B0 243      .SBTTL MAKE_AI_JNL - Put operation specific info in AI jnl
00B0 244
00B0 245
00B0 246 :++
00B0 247 :
00B0 248 : FUNCTIONAL DESCRIPTION:
00B0 249 :
00B0 250 :     MAKE_AI_JNL moves the operation specific information in the journal
00B0 251 :     entry for an AI journal.
00B0 252 :
00B0 253 : CALLING SEQUENCE:
00B0 254 :
00B0 255 :     BSBW    MAKE_AI_JNL
00B0 256 :
00B0 257 : INPUT PARAMETERS:
00B0 258 :
00B0 259 :     R1      Address of record image portion of journal buffer
00B0 260 :     R3      Journal buffer address
00B0 261 :     R4      Journal BDB address
00B0 262 :     R5      Record address
00B0 263 :     R6      Record length
00B0 264 :     R8      RAB
00B0 265 :     R9      IRAB
00B0 266 :     R10     IFAB
00B0 267 :
00B0 268 : IMPLICIT INPUTS:
00B0 269 :
00B0 270 :     None
00B0 271 :
00B0 272 : OUTPUT PARAMETERS:
00B0 273 :
00B0 274 :     R0      Status
00B0 275 :     R1 - R3 Destroyed
00B0 276 :
00B0 277 : IMPLICIT OUTPUTS:
00B0 278 :
00B0 279 :     None
00B0 280 :
00B0 281 : COMPLETION CODES:
00B0 282 :
00B0 283 :     RHB or RBF
00B0 284 :
00B0 285 : SIDE EFFECTS:
00B0 286 :
00B0 287 :     None
00B0 288 :
00B0 289 : --
00B0 290 :
00B0 291 : MAKE_AI_JNL:
00B0 292 :
00B0 293 :
00B0 294 : +
00B0 295 :
00B0 296 : Fill in AI specific information in the journal entry. Then if the record
00B0 297 : being journaled is VFC format, copy the fixed header portion into the
00B0 298 : the record image.
00B0 299 :
```



```
00B0 300 :-  
00B0 301  
14 06 50 30 BB 00B0 302 PUSHR #^M<R4, R5> ; Save jnl bdb and record adr  
A4 A3 01 DO 00B2 303 MOVL #1, R0 ; Assume success  
0048 8F 90 00B5 304 MOVB #RJR$C_RMS_AI, RJR$B_JNL_TYPE(R3) ; This is an AI journal entry  
05 A3 1B 91 00B9 305 MOVW #RJR$C_RECLEN, BDB$W_NUMB(R4) ; Journal entry contains at  
57 13 00BF 306 ; least the overhead  
05 A3 1B 91 00BF 307 CMPB #RJR$C_TRUNCATE, RJR$B_OPER(R3) ; Is the operation truncation?  
57 13 00C3 308 BEQL TRUNC_ENTRY ; No need to copy any data  
46 A3 56 B0 00C5 309  
14 A4 56 A0 00C9 310 MOVW R6, RJR$W_RSIZE(R3) ; Save the record's size  
50 AA 03 91 00CD 311 ADDW2 R6, BDB$W_NUMB(R4) ; Add rec len to jnl entry size  
27 12 00D1 312 CMPB #FAB$C_VFC, IFB$B_RFMORG(R10) ; Is the record VFC format?  
00D3 313 BNEQ 40$ ; No, copy the record  
00D3 314  
00D3 315 ;+  
00D3 316  
00D3 317 ; Copy the fixed header portion of the record.  
00D3 318  
00D3 319 :-  
00D3 320  
52 5F AA 9A 00D3 321 MOVZBL IFB$B_FSZ(R10), R2 ; Get length of fixed hdr part  
14 A4 52 A0 00D7 322 ADDW2 R2, BDB$W_NUMB(R4) ; Count FSZ in jnl entry size  
50 2C A8 DO 00DB 323 MOVL RAB$L_RHB(R8), R0 ; Get adr of user's hdr buffer  
08 12 00DF 324 BNEQ 20$ ; If buffer adr given, copy hdr  
61 52 00 60 00 2C 00E1 325 MOVCS #0, (R0), #0, R2, (R1) ; Zero hdr part of record image  
0B 11 00E7 326 BRB 30$ ; Copy variable portion of rec  
00E9 327 20$: IFNORD R2, (R5), ERRRHB, IRB$B_MODE(R9) ; Quit if hdr can't be read  
61 65 52 28 00F0 328 MOVCS R2, (R5), (R1) ; Copy hdr part to record image  
51 53 DO 00F4 329 30$: MOVL R3, R1 ; Point to next byte in rec image  
54 6E 7D 00F7 330 MOVQ (SP), R4 ; Retrieve BDB and rec adr  
00FA 331  
00FA 332 ;+  
00FA 333  
00FA 334 ; Copy the record to the journal entry.  
00FA 335  
00FA 336 :-  
00FA 337  
0200 8F 56 B1 00FA 338 40$: CMPW R6, #512 ; Is record longer than a page?  
0B 1B 00FF 339 BLEQU 50$ ; Yes, do short read probe  
00000000 EF 16 0101 340 JSB RMS$PROBEREAD ; No, do a long probe  
1C 50 E9 0107 341 BLBC R0, ERRBUF ; Get out on error  
0B 11 010A 342 BRB 60$ ; Continue processing  
010C 343 50$: IFNORD R6, (R5), ERRBUF, IRB$B_MODE(R9) ; Quit if record can't be read  
61 65 56 28 0113 344 MOVCS R6, (R5), (R1) ; Copy record to jnl entry  
0117 345 60$: RMSSUC ; Journal entry is complete  
00 11 011A 346 BRB EXIT_AI_RTN  
011C 347  
011C 348 ;+  
011C 349  
011C 350 ; All the information necessary for AI recovery of a $TRUNCATE operation is  
011C 351 ; already in the RMS journal record (RJR). Therefore, no further modification  
011C 352 ; needs to be done to the journal entry.  
011C 353  
011C 354 :-  
011C 355  
011C 356 TRUNC_ENTRY:
```


30	BA	011C	357	EXIT_AI_RTN:	
	05	011C	358	POPR	#^M<R4, R5>
		011E	359	RSB	
		011F	360		; Return to caller
		011F	361	ERRRHB:	
F6	11	011F	362	RMSERR RHB	
		0124	363	BRB	EXIT_AI_RTN
		0126	364		
		0126	365	ERRBUF:	
EF	11	0126	366	RMSERR RBF	
		012B	367	BRB	EXIT_AI_RTN


```
012D 369
012D 370      .SBTTL MAKE_BI_JNL - Put operation specific info in BI jnl
012D 371
012D 372 :++
012D 373
012D 374      FUNCTIONAL DESCRIPTION:
012D 375
012D 376      MAKE_BI_JNL moves the operation specific information in the journal
012D 377      entry for an BI journal.
012D 378
012D 379      CALLING SEQUENCE:
012D 380
012D 381      BSBW MAKE_BI_JNL
012D 382
012D 383      INPUT PARAMETERS:
012D 384
012D 385      R1      Address of record image portion of journal buffer
012D 386      R3      Journal buffer address
012D 387      R4      Journal BDB address
012D 388      R6      Record length
012D 389      R8      RAB
012D 390      R9      IRAB
012D 391      R10     IFAB
012D 392
012D 393      IMPLICIT INPUTS:
012D 394
012D 395      None
012D 396
012D 397      OUTPUT PARAMETERS:
012D 398
012D 399      R0      Status
012D 400      R1 - R3 Destroyed
012D 401
012D 402      IMPLICIT OUTPUTS:
012D 403
012D 404      None
012D 405
012D 406      COMPLETION CODES:
012D 407
012D 408      Any completion code returned by RM$NXTBLK1
012D 409
012D 410      SIDE EFFECTS:
012D 411
012D 412      None
012D 413
012D 414 :--
012D 415
012D 416
012D 417 MAKE_BI_JNL:
012D 418
012D 419 :+
012D 420
012D 421      Fill in BI/RU specific information in the journal entry.
012D 422
012D 423 :-
012D 424
00F0 8F BB 012D 425      PUSHF  #^M<R4, R5, R6, R7>      ; Save jnl bdb, record adr & len
```



```
14 A4 50 01 D0 0131 426      MOVL #1, R0      ; Assume success
      0048 8F B0 0134 427      MOVW #RJR$C_RECLN, BDB$W_NUMB(R4) ; Journal entry contains at
      05 A3 13 91 013A 428      ; least the overhead
      0C 12 013E 429      CMPB #RJR$_PUT, RJR$B_OPER(R3) ; Is the operation $PUT?
      0169 31 0140 430      BNEQ 20$ ; No, move data to jnl entry
      05 A3 1C 91 0143 431      BRW PUT_ENTRY ; Yes, no need to move data
      03 13 0147 432 10$: CMPB #RJR$_UPDATE, RJR$B_OPER(R3) ; Is the operation $UPDATE?
      0086 31 0149 433      BEQL 20$ ; Yes
      014C 434      BRW BI_TRUNC_ENTRY ; No, it's truncate on put, or $STRUN
      014C 435
      014C 436
      014C 437 ;+
      014C 438 ;+
      014C 439 ; Adjust journal entry size to compensate for any overhead. If it is VFC,
      014C 440 ; include size of fixed header portion. For UDF, VAR, FIX and VFC do not add in
      014C 441 ; size of control (count) field. Do include overhead for STM, STMLF and STMCR.
      014C 442 ; The terminators are counted as overhead, but are also part of the record.
      014C 443 ;+
      014C 444 ;+
      014C 445 ;+
      46 A3 56 B0 014C 446 20$: MOVW R6, RJR$W_RSIZE(R3) ; Save rec size in jnl entry
      0150 447
      0150 448      ASSUME FAB$C_VFC GT FAB$C_UDF
      0150 449      ASSUME FAB$C_VFC GT FAB$C_VAR
      0150 450      ASSUME FAB$C_VFC GT FAB$C_FIX
      0150 451      ASSUME FAB$C_STM GT FAB$C_VFC
      0150 452      ASSUME FAB$C_STMLF GT FAB$C_VFC
      0150 453      ASSUME FAB$C_STMCR GT FAB$C_VFC
      0150 454      ASSUME FAB$C_STMCR EQ FAB$C_MAXRFM
      0150 455
      50 AA 03 91 0150 456      CMPB #FAB$C_VFC, IFB$B_RFMORG(R10) ; Is the record VFC format?
      08 1F 0154 457      BLSSU 30$ ; No, ignore overhead (count field)
      16 1A 0156 458      BGTRU 40$ ; No, include overhead (terminators)
      56 5F AA 80 0158 459      ADDB2 IFB$B_FSZ(R10), R6 ; Yes, include header portion
      10 11 015C 460      BRB 40$
      00A1 CA 02 93 015E 461 30$: BITB #<IFB$V_BI_RECVR!IFB$V_RU_RECVR>, IFB$B_RECVRFLGS(R10) ; If in recov
      09 12 0163 462      BNEQ 40$ ; terminators are already counted
      56 64 A9 A0 0165 463      ADDW2 IRB$W_ROVHDSZ(R9), R6 ; Stream format, include overhead
      46 A3 64 A9 A0 0169 464      ADDW2 IRB$W_ROVHDSZ(R9), RJR$W_RSIZE(R3) ; Add overhead to jnl entry size
      14 A4 56 A0 016E 465 40$: ADDW2 R6, BDB$W_NUMB(R4) ; Increase size of jnl buffer
      0172 466
      0172 467 ;+
      0172 468 ;+
      0172 469 ; Locate the first byte of the data to be copied to the journal entry.
      0172 470 ; NOTE -- This assumes 512 byte blocks.
      0172 471 ;+
      0172 472 ;+
      0172 473 ;+
      50 20 A9 D0 0172 474      MOVL IRB$C_CURBDB(R9), R0 ; Retrieve BDB for buffer
      55 48 A0 9A 0176 475      MOVZBL BDB$B_REL_VBN(R0), R5 ; Get block containing record
      55 48 AA C4 017A 476      MULL2 IFB$C_DEVBUSIZ(R10), R5 ; Convert to byte offset
      55 18 A0 C0 017E 477      ADDL2 BDB$C_ADDR(R0), R5 ; Add offset to buffer address
      50 4C A9 3C 0182 478      MOVZWL IRB$W_RP_OFF(R9), R0 ; Get offset with in block
      55 50 C0 0186 479      ADDL2 R0, R5 ; Point to first byte of record
      0189 480
      0189 481 ;+
      0189 482 ;+
      0189 483 ;+
      0189 484 ;+
      0189 485 ;+
      0189 486 ;+
      0189 487 ;+
      0189 488 ;+
      0189 489 ;+
      0189 490 ;+
      0189 491 ;+
      0189 492 ;+
      0189 493 ;+
      0189 494 ;+
      0189 495 ;+
      0189 496 ;+
      0189 497 ;+
      0189 498 ;+
      0189 499 ;+
      0189 500 ;+
      0189 501 ;+
      0189 502 ;+
      0189 503 ;+
      0189 504 ;+
      0189 505 ;+
      0189 506 ;+
      0189 507 ;+
      0189 508 ;+
      0189 509 ;+
      0189 510 ;+
      0189 511 ;+
      0189 512 ;+
      0189 513 ;+
      0189 514 ;+
      0189 515 ;+
      0189 516 ;+
      0189 517 ;+
      0189 518 ;+
      0189 519 ;+
      0189 520 ;+
      0189 521 ;+
      0189 522 ;+
      0189 523 ;+
      0189 524 ;+
      0189 525 ;+
      0189 526 ;+
      0189 527 ;+
      0189 528 ;+
      0189 529 ;+
      0189 530 ;+
      0189 531 ;+
      0189 532 ;+
      0189 533 ;+
      0189 534 ;+
      0189 535 ;+
      0189 536 ;+
      0189 537 ;+
      0189 538 ;+
      0189 539 ;+
      0189 540 ;+
      0189 541 ;+
      0189 542 ;+
      0189 543 ;+
      0189 544 ;+
      0189 545 ;+
      0189 546 ;+
      0189 547 ;+
      0189 548 ;+
      0189 549 ;+
      0189 550 ;+
      0189 551 ;+
      0189 552 ;+
      0189 553 ;+
      0189 554 ;+
      0189 555 ;+
      0189 556 ;+
      0189 557 ;+
      0189 558 ;+
      0189 559 ;+
      0189 560 ;+
      0189 561 ;+
      0189 562 ;+
      0189 563 ;+
      0189 564 ;+
      0189 565 ;+
      0189 566 ;+
      0189 567 ;+
      0189 568 ;+
      0189 569 ;+
      0189 570 ;+
      0189 571 ;+
      0189 572 ;+
      0189 573 ;+
      0189 574 ;+
      0189 575 ;+
      0189 576 ;+
      0189 577 ;+
      0189 578 ;+
      0189 579 ;+
      0189 580 ;+
      0189 581 ;+
      0189 582 ;+
      0189 583 ;+
      0189 584 ;+
      0189 585 ;+
      0189 586 ;+
      0189 587 ;+
      0189 588 ;+
      0189 589 ;+
      0189 590 ;+
      0189 591 ;+
      0189 592 ;+
      0189 593 ;+
      0189 594 ;+
      0189 595 ;+
      0189 596 ;+
      0189 597 ;+
      0189 598 ;+
      0189 599 ;+
      0189 600 ;+
      0189 601 ;+
      0189 602 ;+
      0189 603 ;+
      0189 604 ;+
      0189 605 ;+
      0189 606 ;+
      0189 607 ;+
      0189 608 ;+
      0189 609 ;+
      0189 610 ;+
      0189 611 ;+
      0189 612 ;+
      0189 613 ;+
      0189 614 ;+
      0189 615 ;+
      0189 616 ;+
      0189 617 ;+
      0189 618 ;+
      0189 619 ;+
      0189 620 ;+
      0189 621 ;+
      0189 622 ;+
      0189 623 ;+
      0189 624 ;+
      0189 625 ;+
      0189 626 ;+
      0189 627 ;+
      0189 628 ;+
      0189 629 ;+
      0189 630 ;+
      0189 631 ;+
      0189 632 ;+
      0189 633 ;+
      0189 634 ;+
      0189 635 ;+
      0189 636 ;+
      0189 637 ;+
      0189 638 ;+
      0189 639 ;+
      0189 640 ;+
      0189 641 ;+
      0189 642 ;+
      0189 643 ;+
      0189 644 ;+
      0189 645 ;+
      0189 646 ;+
      0189 647 ;+
      0189 648 ;+
      0189 649 ;+
      0189 650 ;+
      0189 651 ;+
      0189 652 ;+
      0189 653 ;+
      0189 654 ;+
      0189 655 ;+
      0189 656 ;+
      0189 657 ;+
      0189 658 ;+
      0189 659 ;+
      0189 660 ;+
      0189 661 ;+
      0189 662 ;+
      0189 663 ;+
      0189 664 ;+
      0189 665 ;+
      0189 666 ;+
      0189 667 ;+
      0189 668 ;+
      0189 669 ;+
      0189 670 ;+
      0189 671 ;+
      0189 672 ;+
      0189 673 ;+
      0189 674 ;+
      0189 675 ;+
      0189 676 ;+
      0189 677 ;+
      0189 678 ;+
      0189 679 ;+
      0189 680 ;+
      0189 681 ;+
      0189 682 ;+
      0189 683 ;+
      0189 684 ;+
      0189 685 ;+
      0189 686 ;+
      0189 687 ;+
      0189 688 ;+
      0189 689 ;+
      0189 690 ;+
      0189 691 ;+
      0189 692 ;+
      0189 693 ;+
      0189 694 ;+
      0189 695 ;+
      0189 696 ;+
      0189 697 ;+
      0189 698 ;+
      0189 699 ;+
      0189 700 ;+
      0189 701 ;+
      0189 702 ;+
      0189 703 ;+
      0189 704 ;+
      0189 705 ;+
      0189 706 ;+
      0189 707 ;+
      0189 708 ;+
      0189 709 ;+
      0189 710 ;+
      0189 711 ;+
      0189 712 ;+
      0189 713 ;+
      0189 714 ;+
      0189 715 ;+
      0189 716 ;+
      0189 717 ;+
      0189 718 ;+
      0189 719 ;+
      0189 720 ;+
      0189 721 ;+
      0189 722 ;+
      0189 723 ;+
      0189 724 ;+
      0189 725 ;+
      0189 726 ;+
      0189 727 ;+
      0189 728 ;+
      0189 729 ;+
      0189 730 ;+
      0189 731 ;+
      0189 732 ;+
      0189 733 ;+
      0189 734 ;+
      0189 735 ;+
      0189 736 ;+
      0189 737 ;+
      0189 738 ;+
      0189 739 ;+
      0189 740 ;+
      0189 741 ;+
      0189 742 ;+
      0189 743 ;+
      0189 744 ;+
      0189 745 ;+
      0189 746 ;+
      0189 747 ;+
      0189 748 ;+
      0189 749 ;+
      0189 750 ;+
      0189 751 ;+
      0189 752 ;+
      0189 753 ;+
      0189 754 ;+
      0189 755 ;+
      0189 756 ;+
      0189 757 ;+
      0189 758 ;+
      0189 759 ;+
      0189 760 ;+
      0189 761 ;+
      0189 762 ;+
      0189 763 ;+
      0189 764 ;+
      0189 765 ;+
      0189 766 ;+
      0189 767 ;+
      0189 768 ;+
      0189 769 ;+
      0189 770 ;+
      0189 771 ;+
      0189 772 ;+
      0189 773 ;+
      0189 774 ;+
      0189 775 ;+
      0189 776 ;+
      0189 777 ;+
      0189 778 ;+
      0189 779 ;+
      0189 780 ;+
      0189 781 ;+
      0189 782 ;+
      0189 783 ;+
      0189 784 ;+
      0189 785 ;+
      0189 786 ;+
      0189 787 ;+
      0189 788 ;+
      0189 789 ;+
      0189 790 ;+
      0189 791 ;+
      0189 792 ;+
      0189 793 ;+
      0189 794 ;+
      0189 795 ;+
      0189 796 ;+
      0189 797 ;+
      0189 798 ;+
      0189 799 ;+
      0189 800 ;+
      0189 801 ;+
      0189 802 ;+
      0189 803 ;+
      0189 804 ;+
      0189 805 ;+
      0189 806 ;+
      0189 807 ;+
      0189 808 ;+
      0189 809 ;+
      0189 810 ;+
      0189 811 ;+
      0189 812 ;+
      0189 813 ;+
      0189 814 ;+
      0189 815 ;+
      0189 816 ;+
      0189 817 ;+
      0189 818 ;+
      0189 819 ;+
      0189 820 ;+
      0189 821 ;+
      0189 822 ;+
      0189 823 ;+
      0189 824 ;+
      0189 825 ;+
      0189 826 ;+
      0189 827 ;+
      0189 828 ;+
      0189 829 ;+
      0189 830 ;+
      0189 831 ;+
      0189 832 ;+
      0189 833 ;+
      0189 834 ;+
      0189 835 ;+
      0189 836 ;+
      0189 837 ;+
      0189 838 ;+
      0189 839 ;+
      0189 840 ;+
      0189 841 ;+
      0189 842 ;+
      0189 843 ;+
      0189 844 ;+
      0189 845 ;+
      0189 846 ;+
      0189 847 ;+
      0189 848 ;+
      0189 849 ;+
      0189 850 ;+
      0189 851 ;+
      0189 852 ;+
      0189 853 ;+
      0189 854 ;+
      0189 855 ;+
      0189 856 ;+
      0189 857 ;+
      0189 858 ;+
      0189 859 ;+
      0189 860 ;+
      0189 861 ;+
      0189 862 ;+
      0189 863 ;+
      0189 864 ;+
      0189 865 ;+
      0189 866 ;+
      0189 867 ;+
      0189 868 ;+
      0189 869 ;+
      0189 870 ;+
      0189 871 ;+
      0189 872 ;+
      0189 873 ;+
      0189 874 ;+
      0189 875 ;+
      0189 876 ;+
      0189 877 ;+
      0189 878 ;+
      0189 879 ;+
      0189 880 ;+
      0189 881 ;+
      0189 882 ;+
      0189 883 ;+
      0189 884 ;+
      0189 885 ;+
      0189 886 ;+
      0189 887 ;+
      0189 888 ;+
      0189 889 ;+
      0189 890 ;+
      0189 891 ;+
      0189 892 ;+
      0189 893 ;+
      0189 894 ;+
      0189 895 ;+
      0189 896 ;+
      0189 897 ;+
      0189 898 ;+
      0189 899 ;+
      0189 900 ;+
      0189 901 ;+
      0189 902 ;+
      0189 903 ;+
      0189 904 ;+
      0189 905 ;+
      0189 906 ;+
      0189 907 ;+
      0189 908 ;+
      0189 909 ;+
      0189 910 ;+
      0189 911 ;+
      0189 912 ;+
      0189 913 ;+
      0189 914 ;+
      0189 915 ;+
      0189 916 ;+
      0189 917 ;+
      0189 918 ;+
      0189 919 ;+
      0189 920 ;+
      0189 921 ;+
      0189 922 ;+
      0189 923 ;+
      0189 924 ;+
      0189 925 ;+
      0189 926 ;+
      0189 927 ;+
      0189 928 ;+
      0189 929 ;+
      0189 930 ;+
      0189 931 ;+
      0189 932 ;+
      0189 933 ;+
      0189 934 ;+
      0189 935 ;+
      0189 936 ;+
      0189 937 ;+
      0189 938 ;+
      0189 939 ;+
      0189 940 ;+
      0189 941 ;+
      0189 942 ;+
      0189 943 ;+
      0189 944 ;+
      0189 945 ;+
      0189 946 ;+
      0189 947 ;+
      0189 948 ;+
      0189 949 ;+
      0189 950 ;+
      0189 951 ;+
      0189 952 ;+
      0189 953 ;+
      0189 954 ;+
      0189 955 ;+
      0189 956 ;+
      0189 957 ;+
      0189 958 ;+
      0189 959 ;+
      0189 960 ;+
      0189 961 ;+
      0189 962 ;+
      0189 963 ;+
      0189 964 ;+
      0189 965 ;+
      0189 966 ;+
      0189 967 ;+
      0189 968 ;+
      0189 969 ;+
      0189 970 ;+
      0189 971 ;+
      0189 972 ;+
      0189 973 ;+
      0189 974 ;+
      0189 975 ;+
      0189 976 ;+
      0189 977 ;+
      0189 978 ;+
      0189 979 ;+
      0189 980 ;+
      0189 981 ;+
      0189 982 ;+
      0189 983 ;+
      0189 984 ;+
      0189 985 ;+
      0189 986 ;+
      0189 987 ;+
      0189 988 ;+
      0189 989 ;+
      0189 990 ;+
      0189 991 ;+
      0189 992 ;+
      0189 993 ;+
      0189 994 ;+
      0189 995 ;+
      0189 996 ;+
      0189 997 ;+
      0189 998 ;+
      0189 999 ;+
      0189 1000 ;+
      0189 1001 ;+
      0189 1002 ;+
      0189 1003 ;+
      0189 1004 ;+
      0189 1005 ;+
      0189 1006 ;+
      0189 1007 ;+
      0189 1008 ;+
      0189 1009 ;+
      0189 1010 ;+
      0189 1011 ;+
      0189 1012 ;+
      0189 1013 ;+
      0189 1014 ;+
      0189 1015 ;+
      0189 1016 ;+
      0189 1017 ;+
      0189 1018 ;+
      0189 1019 ;+
      0189 1020 ;+
      0189 1021 ;+
      0189 1022 ;+
      0189 1023 ;+
      0189 1024 ;+
      0189 1025 ;+
      0189 1026 ;+
      0189 1027 ;+
      0189 1028 ;+
      0189 1029 ;+
      0189 1030 ;+
      0189 1031 ;+
      0189 1032 ;+
      0189 1033 ;+
      0189 1034 ;+
      0189 1035 ;+
      0189 1036 ;+
      0189 1037 ;+
      0189 1038 ;+
      0189 1039 ;+
      0189 1040 ;+
      0189 1041 ;+
      0189 1042 ;+
      0189 1043 ;+
      0189 1044 ;+
      0189 1045 ;+
      0189 1046 ;+
      0189 1047 ;+
      0189 1048 ;+
      0189 1049 ;+
      0189 1050 ;+
      0189 1051 ;+
      0189 1052 ;+
      0189 1053 ;+
      0189 1054 ;+
      0189 1055 ;+
      0189 1056 ;+
      0189 1057 ;+
      0189 1058 ;+
      0189 1059 ;+
      0189 1060 ;+
      0189 1061 ;+
      0189 1062 ;+
      0189 1063 ;+
      0189 1064 ;+
      0189 1065 ;+
      0189 1066 ;+
      0189 1067 ;+
      0189 1068 ;+
      0189 1069 ;+
      0189 1070 ;+
      0189 1071 ;+
      0189 1072 ;+
      0189 1073 ;+
      0189 1074 ;+
      0189 1075 ;+
      0189 1076 ;+
      0189 1077 ;+
      0189 1078 ;+
      0189 1079 ;+
      0189 1080 ;+
      0189 1081 ;+
      0189 1082 ;+
      0189 1083 ;+
      0189 1084 ;+
      0189 1085 ;+
      0189 1086 ;+
      0189 1087 ;+
      0189 1088 ;+
      0189 1089 ;+
      0189 1090 ;+
      0189 1091 ;+
      0189 1092 ;+
      0189 1093 ;+
      0189 1094 ;+
      0189 1095 ;+
      0189 1096 ;+
      0189 1097 ;+
      0189 1098 ;+
      0189 1099 ;+
      0189 1100 ;+
      0189 1101 ;+
      0189 1102 ;+
      0189 1103 ;+
      0189 1104 ;+
      0189 1105 ;+
      0189 1106 ;+
      0189 1107 ;+
      0189 1108 ;+
      0189 1109 ;+
      0189 1110 ;+
      0189 1111 ;+
      0189 1112 ;+
      0189 1113 ;+
      0189 1114 ;+
      0189 1115 ;+
      0189 1116 ;+
      0189 1117 ;+
      0189 1118 ;+
      0189 1119 ;+
      0189 1120 ;+
      0189 1121 ;+
      0189 1122 ;+
      0189 1123 ;+
      0189 1124 ;+
      0189 1125 ;+
      0189 1126 ;+
      0189 1127 ;+
      0189 1128 ;+
      0189 1129 ;+
      0189 1130 ;+
      0189 1131 ;+
      0189 1132 ;+
      0189 1133 ;+
      0189 1134 ;+
      0189 1135 ;+
      0189 1136 ;+
      0189 1137 ;+
      0189 1138 ;+
      0189 1139 ;+
      0189 1140 ;+
      0189 1141 ;+
      0189 1142 ;+
      0189 1143 ;+
      0189 1144 ;+
      0189 1145 ;+
      0189 1146 ;+
      0189 1147 ;+
      0189 1148 ;+
      0189 1149 ;+
      0189 1150 ;+
      0189 1151 ;+
      0189 1152 ;+
      0189 1153 ;+
      0189 1154 ;+
      0189 1155 ;+
      0189 1156 ;+
      0189 1157 ;+
      0189 1158 ;+
      0189 1159 ;+
      0189 1160 ;+
      0189 1161 ;+
      0189 1162 ;+
      0189 1163 ;+
      0189 1164 ;+
      0189 1165 ;+
      0189 1166 ;+
      0189 1167 ;+
      0189 1168 ;+
      0189 1169 ;+
      0189 1170 ;+
      0189 1171 ;+
      0189 1172 ;+
      0189 1173 ;+
      0189 1174 ;+
      0189 1175 ;+
      0189 1176 ;+
      0189 1177 ;+
      0189 1178 ;+
      0189 1179 ;+
      0189 1180 ;+
      0189 1181 ;+
      0189 1182 ;+
      0189 1183 ;+
      0189 1184 ;+
      0189 1185 ;+
      0189 1186 ;+
      0189 1187 ;+
      0189 1188 ;+
      0189 1189 ;+
      0189 1190 ;+
      0189 1191 ;+
      0189 1192 ;+
      0189 1193 ;+
      0189 1194 ;+
      0189 1195 ;+
      0189 1196 ;+
      0189 1197 ;+
      0189 1198 ;+
      0189 1199 ;+
      0189 1200 ;+
      0189 1201 ;+
      0189 1202 ;+
      0189 1203 ;+
      0189 1204 ;+
      0189 1205 ;+
      0189 1206 ;+
      0189 1207 ;+
      0189 1208 ;+
      0189 1209 ;+
      0189 1210 ;+
      0189 1211 ;+
      0189 1212 ;+
      0189 1213 ;+
      0189 1214 ;+
      0189 1215 ;+
      0189 1216 ;+
      0189 1217 ;+
      0189 1218 ;+
      0189 1219 ;+
      0189 1220 ;+
      0189 1221 ;+
      0189 1222 ;+
      0189 1223 ;+
      0189 1224 ;+
      0189 1225 ;+
      0189 1226 ;+
      0189 1227 ;+
      0189 1228 ;+
      0189 1229 ;+
      0189 1230 ;+
      0189 1231 ;+
      0189 1232 ;+
      0189 1233 ;+
      0189 1234 ;+
      0189 1235 ;+
      0189 1236 ;+
      0189 1237 ;+
      0189 1238 ;+
      0189 1239 ;+
      0189 1240 ;+
      0189 1241 ;+
      0189 1242 ;+
      0189 1243 ;+
      0189 1244 ;+
      0189 1245 ;+
      0189 1246 ;+
      0189 1247 ;+
      0189 1248 ;+
      0189 1249 ;+
      0189 1250 ;+
      0189 1251 ;+
      0189 1252 ;+
      0189 1253 ;+
      0189 1254 ;+
      0189 1255 ;+
      0189 1256 ;+
      0189 1257 ;+
      0189 1258 ;+
      0189 1259 ;+
      0189 1260 ;+
      0
```



```
0189 483 : If there is a count field preceeding the record, skip over it so that we are
0189 484 : truly pointing to the first byte of the record. Since the total record size
0189 485 : includes the count field, if that value is different from the one calculated
0189 486 : for the journal entry, then the record has a count field and it should be
0189 487 : skipped.
0189 488 :
0189 489 :-
0189 490 :
50 66 A9 3C 0189 491      MOVZWL  IRB$W_RTOTLSZ(R9), R0      ; Get total record size
50 50 56 C2 018D 492      SUBL2   R6, R0              ; Determine count field length
55 50 C0 0190 493      ADDL2   R0, R5              ; Move pointer over count field
0193 494 :
0193 495 :+
0193 496 :
0193 497 : Save the current record pointer, in case the record crosses into the next
0193 498 : buffer causing the rest of the record is read into the buffer. After the
0193 499 : whole record has been copied to the journal entry, the current record pointer
0193 500 : will be needed to restore the current contents of the buffer.
0193 501 :
0193 502 :-
0193 503 :
7E 48 A9 7D 0193 504      ASSUME  IRB$W_RP_OFF EQ <IRB$L_RP_VBN + 4>
0193 505      MOVQ   IRB$L_RP_VBN(R9), -(SP)
0197 506 :
0197 507 :+
0197 508 :
0197 509 : Copy the record to the journal entry. The current register contents are:
0197 510 :
0197 511 : R1 - address of first byte of RJR record image (destination)
0197 512 : R5 - first byte of record in buffer (source)
0197 513 : R6 - number of bytes to transfer to journal entry
0197 514 : R7 - end of buffer address + 1
0197 515 :
0197 516 :-
0197 517 :
0197 518 COPY_DATA:
50 57 55 C3 0197 519      SUBL3   R5, R7, R0      ; Get # of bytes left in source buff
56 50 D1 019B 520      CMPL   R0, R6      ; Is whole record in buffer?
56 03 1B 019E 521      BLEQU  10$, R6      ; No, transfer size = remaining buff
50 56 D0 01A0 522      MOVL   R6, R0      ; Yes, use rec len as transfer size
56 50 C2 01A3 523 10$:  SUBL2   R0, R6      ; Adjust size of record
61 65 50 28 01A6 524      MOVC3  R0, (R5), (R1) ; Copy the (partial) record
56 D5 01AA 525      TSTL   R6      ; Any data left to copy?
02 12 01AC 526      BNEQ   20$, R6      ; Yes, refill buffer, copy rest of r
15 11 01AE 527      BRB    RESTORE_BUFF ; No, copy is complete
08 BB 01B0 528 20$:  PUSHR  #^M<R3> ; Save source and destination
00FC 30 01B2 529      BSBW   CHANGE_BUFF ; Get next buffer
55 51 D0 01B5 530      MOVL   R1, R5      ; Save source location
51 8ED0 01B8 531      POPL   R1      ; Restore the destination
D9 50 E8 01BB 532      BLBS   R0, COPY_DATA ; Copy rest of record or fall thru t
01BE 533 :
01BE 534 BI_ERROR_EXIT:
01BE 535      ASSUME  IRB$W_RP_OFF EQ <IRB$L_RP_VBN + 4>
48 A9 8E 7D 01BE 536      MOVQ   (SP)+, IRB$L_RP_VBN(R9) ; Retrieve record pointer
00E7 31 01C2 537      BRW    EXIT_BI_RTN ; Return with error status
01C5 538 :
01C5 539 RESTORE_BUFF:
```


40 A9 8E	7D 01C5 540	ASSUME	IRBSW_NRP OFF EQ <IRBSL_NRP_VBN + 4>	
00000000 EF	16 01C5 541	MOVQ	(SP)+, IRBSL_NRP_VBN(R9)	; Retrieve record pointer
00DA	31 01C9 542	JSB	RM\$GETBLKNRP	; Restore contents of the buffer
	31 01CF 543	BRW	EXIT_BI_RTN	; Return with error status


```
01D2 545 :+
01D2 546 :
01D2 547 : The current operation involves truncation ($TRUNCATE or $PUT, with TPT set).
01D2 548 : The rest of the file must be copied to the journal. This is done one buffer
01D2 549 : at a time, from the current VBN to the EOF. The journal entries are formatted
01D2 550 : as BLOCK I/O entries, not record entries. Recovery should be done as a series
01D2 551 : of $WRITES.
01D2 552 :
01D2 553 :
01D2 554 :
01D2 555 :
01D2 556 : *****
01D2 557 : *
01D2 558 : * THE CODE FOR BI JOURNALING OF TRUNCATE OPERATIONS HAS NOT
01D2 559 : * BEEN TESTED.
01D2 560 : *
01D2 561 : *****
01D2 562 :
01D2 563 BI_TRUNC_ENTRY:
01D2 564 :
01D2 565 :
01D2 566 : Write all dirty buffers out to the disk, to be sure that the file is in a
01D2 567 : consistent state before any data is copied to the journal.
01D2 568 :
01D2 569 :
00000000'EF 16 01D2 570 JSB RMS$FLUSH ; Write buffers to disk
3F 50 E8 01D2 571 BLBS R0, 10$ ; If that worked, keep going
00CE 31 01DB 572 BRW EXIT_BI_RTN ; Get out on error
01DE 573 :
01DE 574 :
01DE 575 : Determine the maximum size for the journal entry. Check to see if it fits
01DE 576 : in the current journal buffer.
01DE 577 :
01DE 578 :
55 55 A9 9A 01DE 579 MOVZBL IRB$B_MBC(R9), R5 ; at most, MBC # of blks will be cop
55 55 55 D6 01E2 580 INCL R5 ; MBC is zero based, not one based
55 48 AA C4 01E4 581 MULL2 IFB$L_DEVBUFFSIZ(R10), R5 ; Convert to bytes
55 00000044 8F C0 01E8 582 ADDL #RJR$C_BLKLEN, R5 ; Include jnl entry overhead in size
55 55 2C A4 B1 01EF 583 CMPW BDB$W_ALLOC_SIZE(R4), R5 ; Will it fit in the curr buff?
2D 1E 01F3 584 BGEQU 20$ ; Yes, continue processing
01F5 585 :
01F5 586 :
01F5 587 : Get a new journal buffer is needed; the current one is too small. Initialize
01F5 588 : any journal entry fields which are assumed to already have values in them.
01F5 589 :
01F5 590 :
55 55 DD 01F5 591 PUSHL R5 ; Save jnl buff size
00000000'EF 16 01F7 592 JSB RMS$RETJNLBDB ; Release this buffer
55 000001FF 8F C0 01FD 593 ADDL2 #511, R5 ; Round the number of bytes to
55 000001FF 8F CA 0204 594 BICL #511, R5 ; up to a page boundary
55 55 8ED0 020B 595 POPL R5 ; Restore jnl buffer size
00000000'EF 16 020E 596 JSB RMS$ALDJNLBUF ; Get a new BDB and buffer
03 50 E8 0214 597 BLBS R0, 10$ ; Continue if new BDB is okay
0092 31 0217 598 BRW EXIT_BI_RTN
30 A9 54 D0 021A 599 10$: MOVL R4, IRB$L_JNLBDB(R9) ; Save the jnl BDB address
04 A3 00 90 021E 600 MOVB #RJR$C_SEQ, RJR$B_ORG(R3) ; File is sequential organization
0222 601
```



```
0222 602 :  
0222 603 : Initialize the journal BDB and the journal entry. The jnl entry should look  
0222 604 : like a BLOCK I/O operation is happening.  
0222 605 :  
0222 606 :  
55 1C A4 48 AA D0 0222 607 20$: MOVL IRB$LP_VBN(R10), BDB$L_VBN(R4); Start VBN is VBN of curr rec  
00000044 8F C2 0227 608 : SUBL2 #RJR$C_BLKLEN, R5 ; Ovrhd not included in # bytes to j  
14 A4 55 B0 022E 609 : MOVW R5, BDB$W_NUMB(R4) ; Size of tranfer into jnl entry  
03 A3 03 90 0232 610 : MOVB #RJR$C_BLOCK, RJR$B_ENTRY_TYPE(R3) ; Block mode I/O  
05 A3 1E 90 0236 611 : MOVB #RJR$WRITE, RJR$B_OPER(R3) ; Operation is psuedo-$WRITE  
3C A3 1C A4 D0 023A 612 : MOVL BDB$L_VBN(R4), RJR$L_BLOCK_VBN(R3) ; VBN of 1st blk being jnl'd  
40 A3 14 A4 3C 023F 613 : MOVZWL BDB$W_NUMB(R4), RJR$L_BLOCK_SIZE(R3) ; # of bytes being jnl'd  
0244 614 :  
0244 615 :+  
0244 616 :  
0244 617 : Do until beyond EOF:  
0244 618 : If EOF is in current buffer, set the number of bytes to journal so that  
0244 619 : only data up to the first free byte is read into the journal buffer.  
0244 620 : Read data into the journal entry and write the entry to the journal.  
0244 621 : Determine the start VBN for the next buffer.  
0244 622 :  
0244 623 :-  
0244 624 :  
51 55 A9 9A 0244 625 : MOVZBL IRB$B_MBC(R9), R1 ; EOF is in buffer if:  
51 51 D6 0248 626 : INCL R1 ; (MBC + 1) + start VBN  
52 51 48 A9 C1 024A 627 : ADDL3 IRB$LP_VBN(R9), R1, R2 ; is greater than EBK  
024F 628 MAKE_TRUNC_ENTRY:  
74 AA 52 D1 024F 629 : CMPL R2, IFB$L_EBK(R10) ; Is EOF in the current buffer?  
OF 1F 0253 630 : BLSSU 10$ ; No, journal whole buffer  
50 48 AA 5C AA A3 0255 631 : SUBW3 IFB$W_FFB(R10), IFB$L_DEVBUFFSIZ(R10), R0 ; How many bytes are unused  
14 A4 50 A2 025B 632 : SUBW2 R0, BDB$W_NUMB(R4) ; Decrement # of bytes to jnl  
40 A3 14 A4 3C 025F 633 : MOVZWL BDB$W_NUMB(R4), RJR$L_BLOCK_SIZE(R3) ; Same for jnl entry size  
0264 634 :  
0264 635 :  
0264 636 : Read VBNs into jnl buffer from the disk.  
0264 637 :  
0264 638 :  
18 A4 00000044 8F C0 0264 639 : ASSUME RJR$C_BLKLEN EQ RJR$T_BLOCK  
OE BB 026C 640 10$: ADDL2 #RJR$C_BLKLEN, BDB$L_ADDR(R4) ; Use RJR$T_BLOCK as dest for read  
00000000 EF 16 026E 641 : PUSHF #^M<R1, R2, R3> ; Save pointers and counters  
18 A4 00000044 8F C2 0274 642 : JSB RMSRDBUFWT ; Read in data and wait for completi  
05 50 E8 027C 643 : SUBL2 #RJR$C_BLKLEN, BDB$L_ADDR(R4) ; Return to real start of jnl buffer  
OE BA 027F 644 : BLBS R0, 20$ ; If read worked, continue  
0028 31 0281 645 : POPR #^M<R1, R2, R3> ; Otherwise, restore regs  
0284 646 : BRW EXIT_BI_RTN ; Get our on error  
0284 647 :  
0284 648 :  
0284 649 : Write journal entry out to journal.  
0284 650 :  
0284 651 :  
14 A4 0044 8F A0 0284 652 20$: ADDW2 #RJR$C_BLKLEN, BDB$W_NUMB(R4) ; Ovrhd included in jnl entry size  
2E 10 028A 653 : BSBB WRTBIJNL ; Write jnl entry  
OE BA 028C 654 : POPR #^M<R1, R2, R3> ; Restore pointers and counters  
1B 50 E9 028E 655 : BLBC R0, EXIT_BI_RTN ; Get out on error  
14 A4 0044 8F A2 0291 656 : SUBW2 #RJR$C_BLKLEN, BDB$W_NUMB(R4) ; Remove ovrhd from jnl entry size  
0297 657 :  
0297 658 :
```



```
0297 659 ; Determine start and end VBN of next buffer.
0297 660 ;
0297 661 ;
1C A4 52 D0 0297 662      MOVL      R2, BDB$$_VBN(R4)      ; Start VBN was already calculated
3C A3 1C A4 D0 029B 663      MOVL      BDB$$_VBN(R4), RJR$$_BLOCK_VBN(R3) ; Save start VBN in jnl entry
52 51 C0 02A0 664      ADDL2      R1, R2      ; Get 1st VBN past next buffer
74 AA 52 B1 02A3 665      CMPW      R2, IFB$$_EBK(R10) ; Is EOF in next buffer?
03 1A 02A7 666      BGTRU      EXIT_BI_RTN ; No, do not jnl past EOF (it has be
FFA3 31 02A9 667      BRW      MAKE_TRUNC_ENTRY ; Journal next set of blocks
02AC 668
02AC 669
02AC 670 PUT_ENTRY:
02AC 671 EXIT_BI_RTN:
00F0 8F BA 02AC 672      POPR      #^M<R4, R5, R6, R7>
05 02B0 673
02B1 674      RSB
; Return to caller
```



```
02B1 676 .SBTTL CHANGE_BUFF - get next buffer
02B1 677
02B1 678 :++
02B1 679
02B1 680 : FUNCTIONAL DESCRIPTION:
02B1 681
02B1 682 : CHANGE_BUFF calls RMS$NXTBLK1 for MAKE_BI_JNL.
02B1 683
02B1 684 : CALLING SEQUENCE:
02B1 685
02B1 686 : BSBB CHANGE_BUFF
02B1 687
02B1 688 : INPUT PARAMETERS:
02B1 689
02B1 690 : R8 RAB
02B1 691 : R9 IRAB
02B1 692 : R10 IFAB
02B1 693
02B1 694 : IMPLICIT INPUTS:
02B1 695
02B1 696 : None
02B1 697
02B1 698 : OUTPUT PARAMETERS:
02B1 699
02B1 700 : R0 Status
02B1 701 : R1 - R3 Destroyed
02B1 702
02B1 703 : IMPLICIT OUTPUTS:
02B1 704
02B1 705 : R1 address of current block in buffer
02B1 706 : R7 address of end of buffer + 1
02B1 707
02B1 708 : COMPLETION CODES:
02B1 709
02B1 710 : Any completion code returned by RMS$NXTBLK1
02B1 711
02B1 712 : SIDE EFFECTS:
02B1 713
02B1 714 : None
02B1 715
02B1 716 :--
02B1 717
02B1 718 CHANGE_BUFF:
02B1 719 CLRL R3
02B3 720 JSB RMS$NXTBLK1
02B9 721 RSB
02BA 722
```

00000000 53 D4 02B1 719 ; Indicate read required
EF 16 02B3 720 ; Get new buffer contents
05 02B9 721
02BA 722


```
02BA 724 .SBTTL WRTBIJNL - writes BI/RU journal entry
02BA 725
02BA 726 :++
02BA 727
02BA 728 : FUNCTIONAL DESCRIPTION:
02BA 729
02BA 730 : WRTBIJNL writes a BI/RU jnl entry
02BA 731
02BA 732 : CALLING SEQUENCE:
02BA 733
02BA 734 : BSBB WRTBIJNL
02BA 735
02BA 736 : INPUT PARAMETERS:
02BA 737
02BA 738 : R4 Journal BDB
02BA 739 : R8 RAB
02BA 740 : R9 IRAB
02BA 741 : R10 IFAB
02BA 742
02BA 743 : IMPLICIT INPUTS:
02BA 744
02BA 745 : None
02BA 746
02BA 747 : OUTPUT PARAMETERS:
02BA 748
02BA 749 : R0 Status
02BA 750 : R1 Destroyed
02BA 751
02BA 752 : IMPLICIT OUTPUTS:
02BA 753
02BA 754 : None
02BA 755
02BA 756 : COMPLETION CODES:
02BA 757
02BA 758 : Any completion code returned by RMSWRTJNL
02BA 759
02BA 760 : SIDE EFFECTS:
02BA 761
02BA 762 : None
02BA 763
02BA 764 :--
02BA 765
02BA 766 WRTBIJNL:
12 00A0 CA 02 E1 02BA 767 BBC #IFB$V_BI, IFB$B_JNLFLG(R10), 10$ ; If BI/RU jnl'ing, write a BI/RU
06 A3 02 90 02BA 768 MOVB #RJR$C_RMS_BI, RJR$B_JNL_TYPE(R3) ; This is a BI journal entry
7E 02 9A 02C6 770 PUSH R4 ; Use jnl BDB as relate BDB
FD34' 30 02C9 771 MOVZBL #CJF$ BI, -(SP) ; Pass jnl type to WRTBIJNL
5E 08 C0 02CC 772 BSBW RMSWRTJNL ; Write jnl entry
15 50 E9 02CF 773 ADDL2 #8, SP ; Remove args from stack
02D2 774 BLBC R0, 20$ ; Get out on error
OF 00A0 CA 01 E1 02D2 775 10$: BBC #IFB$V_RU, IFB$B_JNLFLG(R10), 20$ ; If RU jnl'ing, write a RU entry
06 A3 03 90 02D8 776 MOVB #RJR$C_RMS_RU, RJR$B_JNL_TYPE(R3) ; This is an RU journal entry
7E 01 9A 02DC 777 PUSH R4 ; Use jnl BDB as relate BDB
FD1C' 30 02DE 778 MOVZBL #CJF$ RU, -(SP) ; Pass jnl type to WRTBIJNL
5E 08 C0 02E1 779 BSBW RMSWRTJNL ; Write jnl entry
02E4 780 ADDL2 #8, SP ; Remove args from stack
```


RM1JOURNAL
V04-000

Sequential specific journaling C 2
WRTBIJNL - writes BI/RU journal entry

16-SEP-1984 00:50:14 VAX/VMS Macro V04-00
5-SEP-1984 16:23:28 [RMS.SRC]RM1JOURNAL.MAR;1

Page 18
(9)

05	02E7	781		
	02E7	782	20\$:	RSB
	02E8	783		
	02E8	784		.END

RM1JOURNL
Symbol table

Sequential specific journaling

D 2

16-SEP-1984 00:50:14 VAX/VMS Macro V04-00
5-SEP-1984 16:23:28 [RMS.SRC]RM1JOURNL.MAR;1

Page 19
(9)

\$\$PSECT_EP	=	00000000		
\$\$RMSTEST	=	0000001A		
\$\$RMS_PBUGCHK	=	00000010		
\$\$RMS_TBUGCHK	=	00000008		
\$\$RMS_UMODE	=	00000004		
BDB\$B_FLGS	=	0000000A		
BDB\$B_REL_VBN	=	00000048		
BDB\$L_ADDR	=	00000018		
BDB\$L_VBN	=	0000001C		
BDB\$V_PRM	=	00000003		
BDB\$W_ALLOC_SIZE	=	0000002C		
BDB\$W_NUMB	=	00000014		
BI_ERROR_EXIT		000001BE	R	01
BI_TRUNC_ENTRY		000001D2	R	01
CHANGE_BUFF		000002B1	R	01
CJFS_AI	=	00000003		
CJFS_BI	=	00000002		
CJFS_RU	=	00000001		
COPY_DATA		00000197	R	01
ERRBUF		00000126	R	01
ERRRHB		0000011F	R	01
EXIT		000000AD	R	01
EXIT_AI_RTN		0000011C	R	01
EXIT_BI_RTN		000002AC	R	01
FAB\$C_FTX	=	00000001		
FAB\$C_MAXRFM	=	00000006		
FAB\$C_STM	=	00000004		
FAB\$C_STMCR	=	00000006		
FAB\$C_STMLF	=	00000005		
FAB\$C_UDF	=	00000000		
FAB\$C_VAR	=	00000002		
FAB\$C_VFC	=	00000003		
IFB\$B_FSZ	=	0000005F		
IFB\$B_JNLFLG	=	000000A0		
IFB\$B_RECVRFLGS	=	000000A1		
IFB\$B_RFMORG	=	00000050		
IFB\$L_DEVBUSIZ	=	00000048		
IFB\$L_EBK	=	00000074		
IFB\$V_AI	=	00000003		
IFB\$V_BI	=	00000002		
IFB\$V_BI_RECVR	=	00000002		
IFB\$V_RU	=	00000001		
IFB\$V_RU_RECVR	=	00000000		
IFB\$W_FFB	=	0000005C		
IRB\$B_MBC	=	00000055		
IRB\$B_MODE	=	0000000A		
IRB\$L_CURBDB	=	00000020		
IRB\$L_JNLBDB	=	00000030		
IRB\$L_NRP_VBN	=	00000040		
IRB\$L_RP_OFF	=	0000004C		
IRB\$L_RP_VBN	=	00000048		
IRB\$W_NRP_OFF	=	00000044		
IRB\$W_ROVRDSZ	=	00000064		
IRB\$W_RP_OFF	=	0000004C		
IRB\$W_RTOTLSZ	=	00000066		
MAKE_AI_JNL		00000080	R	01
MAKE_BI_JNL		0000012D	R	01

MAKE_TRUNC_ENTRY		0000024F	R	01
PUT_ENTRY		000002AC	R	01
RAB\$L_RHB	=	0000002C		
RESTORE_BUFF		000001C5	R	01
RJR\$B_ENTRY_TYPE	=	00000003		
RJR\$B_JNL_TYPE	=	00000006		
RJR\$B_OPER	=	00000005		
RJR\$B_ORG	=	00000004		
RJR\$C_BLKLEN	=	00000044		
RJR\$C_BLOCK	=	00000003		
RJR\$C_RECLEN	=	00000048		
RJR\$C_RECORD	=	00000002		
RJR\$C_RMS_AI	=	00000001		
RJR\$C_RMS_BI	=	00000002		
RJR\$C_RMS_RU	=	00000003		
RJR\$C_SEQ	=	00000000		
RJR\$L_BLOCK_SIZE	=	00000040		
RJR\$L_BLOCK_VBN	=	0000003C		
RJR\$L_RFA0	=	00000040		
RJR\$T_BLOCK	=	00000044		
RJR\$T_RIMAGE	=	00000048		
RJR\$W_RFA4	=	00000044		
RJR\$W_RSIZE	=	00000046		
RJR\$ PUT	=	00000013		
RJR\$ TPT	=	0000001F		
RJR\$ TRUNCATE	=	0000001B		
RJR\$ UPDATE	=	0000001C		
RJR\$ WRITE	=	0000001E		
RM\$ACDJNLBUF		*****	X	01
RM\$FLUSH		*****	X	01
RM\$GETBLKNRP		*****	X	01
RM\$NXTBLK1		*****	X	01
RM\$PROBEREAD		*****	X	01
RM\$RDBUFWT		*****	X	01
RM\$RETJNLBDB		*****	X	01
RM\$SEQJNL		00000000	RG	01
RM\$WRTJNL		*****	X	01
RM\$ RBF	=	00018654		
RM\$ RHB	=	0001866C		
TRUNC_ENTRY		0000011C	R	01
WRTBIJNL		000002BA	R	01

! Psect synopsis !

PSECT name	Allocation	PSECT No.	Attributes																	
ABS	00000000 (0.)	00 (0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE							
RMSRMS_JOURNAL	000002E8 (744.)	01 (1.)	PIC	USR	CON	REL	GBL	NOSHR	EXE	RD	NOWRT	NOVEC	BYTE							
\$AB\$\$	00000000 (0.)	02 (2.)	NOPIC	USR	CON	ABS	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE							

! Performance indicators !

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.08	00:00:00.75
Command processing	116	00:00:00.64	00:00:04.98
Pass 1	351	00:00:12.00	00:00:36.97
Symbol table sort	0	00:00:01.78	00:00:03.05
Pass 2	138	00:00:02.85	00:00:07.67
Symbol table output	13	00:00:00.11	00:00:00.20
Psect synopsis output	1	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	650	00:00:17.48	00:00:53.64

The working set limit was 1650 pages.
70414 bytes (138 pages) of virtual memory were used to buffer the intermediate code.
There were 70 pages of symbol table space allocated to hold 1275 non-local and 23 local symbols.
784 source lines were read in Pass 1, producing 14 object records in Pass 2.
23 pages of virtual memory were used to define 22 macros.

! Macro library statistics !

Macro library name	Macros defined
\$255\$DUA28:[RMS.OBJ]RMS.MLB;1	12
\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	1
\$255\$DUA28:[SYSLIB]STARLET.MLB;2	5
TOTALS (all libraries)	18

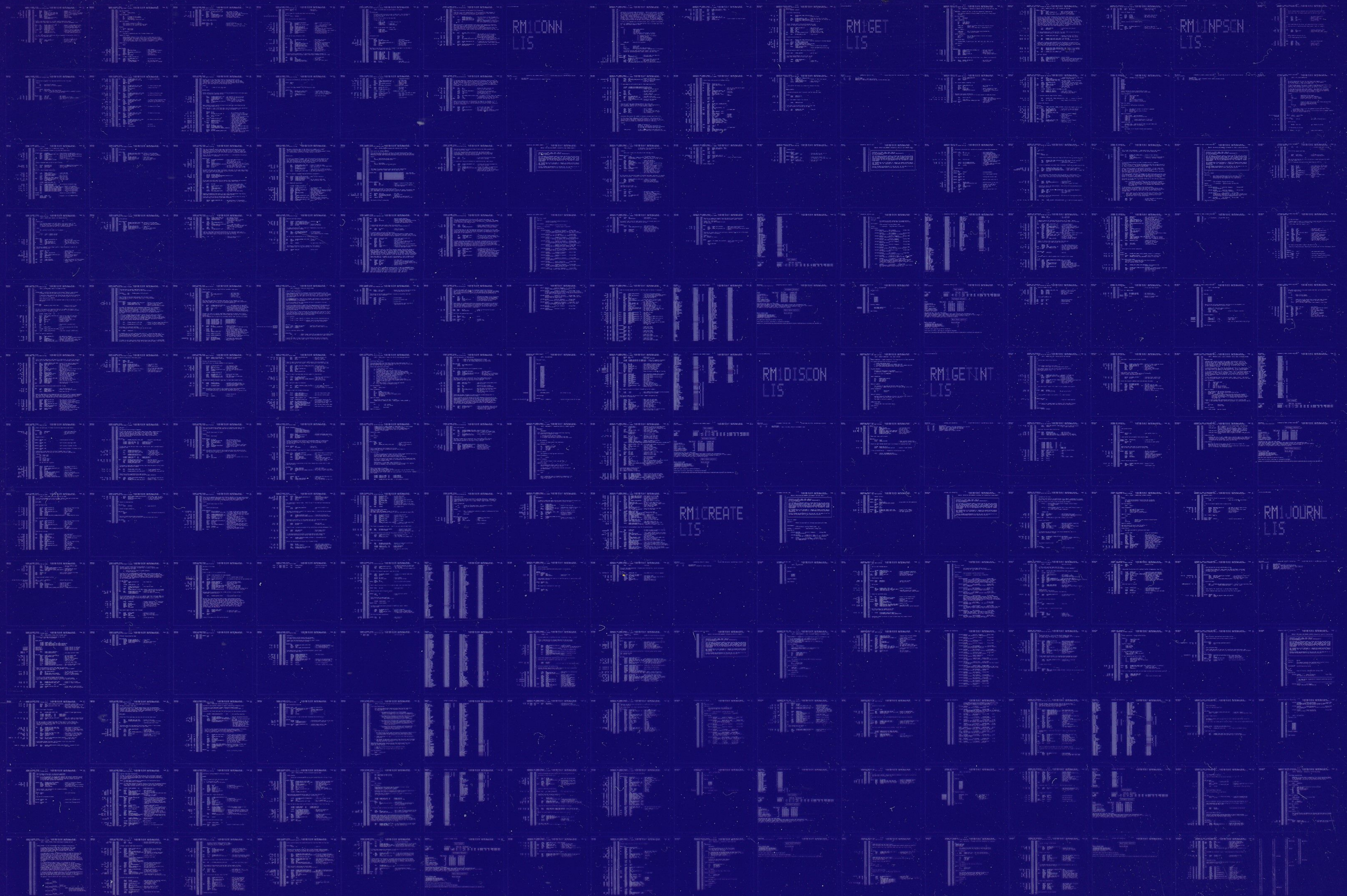
1385 GETs were required to define 18 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LISS:RM1JOURNAL/OBJ=OBJ\$:RM1JOURNAL MSRC\$:RM1JOURNAL/UPDATE=(ENH\$:RM1JOURNAL)+EXECML\$/LIB+LIB\$:RMS/LIB

0321 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



0322

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY